AN EFFICIENT 2D DIFFIE - HELLMAN BASED TWO-SERVER PASSWORD-ONLY AUTHENTICATED KEY EXCHANGE PROTOCOL
Anitha Kumari K1Sudha Sadasivam G2Rohini L3
Address for Correspondence
1. Assistant Professor [Sr. Gr], 3. PG Scholar, Dept. of IT, PSG College of Technology, Peelamedu, Coimbatore 641004
2. Professor, Dept. of CSE, PSG College of Technology, Peelamedu, Coimbatore 641004

ABSTRACT
In emerging technological world, distributed systems such as storage area networks, cloud computing, and Content Delivery Networks (CDNs) are at the forefront for storing, processing and accessing information. However, security is a highest challenge in these large scale distributed systems. To address these issues, we have proposed the first demonstrable secure triangle (2D) based two-server Password Authenticated and Key Exchange (PAKE) protocol with formal proof of security by using the properties of trigonometry. The phenomenal aspect of the proposed protocol is, it is not possible to attain the password from the stored information even under the act of compromising both the servers. In contrast, all existing two server PAKE protocols assume that two servers never conspire. In this paper, we discussed the triangle properties that fuse well with Diffie-Hellman key exchange protocol to guarantee its security against offline dictionary attacks. 2D PAKE protocol is proved mathematically to show the vigor against attacks happening deviously. Proofs and result analysis apparently state that it is infeasible to find out the key and the password by a passive / active adversary.

KEYWORDS: 2D PAKE protocol, Triangle property analysis, Diffie-Hellman key exchange.

1. INTRODUCTION
On a large scale distributed system, a fundamental concern is about authenticating both the local and remote entities. In terms of security, asymmetric key schemes are more stringent than symmetric key schemes by involving public and private keys. A most widely used public key verification scheme is a digital signature algorithm. Unfortunately, an inherent limitation of this scheme is, if the private key used in the certificate is compromised, then the environment becomes insecure. An alternate solution is PAKE protocol where key creation, verification and key exchange operations are carried out based upon the knowledge of pre-stored information. In a single server scenario, the server is liable to attack; when stored credentials fall into the hands of an adversary. In case of the multi-server model, for verification of a user it depends upon multiple servers. To strike a balance, two-server model has been proposed. In addition, it does not involve Public Key Infrastructure (PKI). In this paper, we are proposing a resilient triangle (2D) based two-server PAKE protocol, which is robust against passive and active attacks. 2D protocol assures that determining the key and obtaining a password from the stored information is impossible by a passive / active adversary.

In Section 2 a prelude to related work are discussed, Section 3 elaborates the proposed methodology, Section 4 discusses the security analysis of 2D PAKE, Performance analysis is carried out in Section 5, and conclusion is presented in Section 6.

2. RELATED WORK
The first efficient PAKE protocol was proposed by Bellovin et al. as Encrypted Key Exchange (EKE) methods [4]. It is a scenario in which key exchange engrosses both symmetric and asymmetric encryption. An important concern is susceptible to replay attacks and lack of proof of security. Primary motto is to safeguard users weak passwords against dictionary attacks. In this scenario, challenge response system is used for validating cryptographic keys. It reveals no information either concerning the secret key or public key. For mutual authentication, a Three Party EKE protocol (3PEKE) [17] is proposed where a trusted server gleans password from clients. Intricacy involved in 3PEKE protocol is the requirement of PKI to share a common secret between every pair of clients, which is often not reasonable. Also, it is not resilient to undetectable online guessing attacks and offline password guessing attacks [24]. In 3PEKE protocol the server hoards public-key to circumvent both off-line and undetectable on-line password guessing attacks. The main constraint of this protocol is entangling the user to draw the key and authenticate the public-key of the server. In Gateway-oriented Password basedAuthenticated Key Exchange (GPAKE) scheme [13], the operation between a client and an authentication server is carried out through a gateway. But a malicious gateway is more like an entry point to various attacks. To avoid this threat, Message Authentication Code (MAC) [13] is proposed, to exchange messages safely between client and server. In GPAKE [14] a secret communication channel exists between the gateway and the server. Also, it makes use of a pre-shared password. In threshold password authenticated key exchange [18] servers hold a pile of known public keys. Only if it meets a level of threshold of servers, authentication is ensured. Communicating all the servers demands higher communication bandwidth.

Yang et al. proposed a practical two-server PAKE model [23] in 2006. In this protocol a communication channel is framed between a Service Server (SS) and a Control Server (CS). An active adversary pretending to be a CS can glean the password and key. Subsequently, Lee et al. [16] proposed a two-server PAKE protocol, which is less competent when compared to Yang et al. in terms of performance. An efficient password based two-server authentication and pre-shared key exchange system is
verified with the assistance of smart cards [3] by Chouksey et al. The protocol is strenuous against offline dictionary attacks, replay attack, malicious server attack and man-in-the-middle attack. Procuring and using the device is the major limitation of this approach. Dynamic identity based authentication protocol [19] for two-server architecture frequently uses nonce, one-way hash function and XOR operations. Computational complexity has been reduced compared with other protocols [9], [15], and [23]. ElGamal based encryption and public/secret-key pair generation is proposed by Yi et al. [21] to furnish a high level of security. Using gateways is quite expensive in Yi et al. protocol, provided that it is sturdy against both passive and active attacks.

3. PROPOSED METHODOLOGY

2D PAKE mechanism is designed based on balanced PAKE model where password based variant is a part of session key. It surmounts the drawback of Yang et al. protocol of deriving the password and key. Mathematical research is increasing in the technology world by prompting a related change in the discipline of information security. At this point, we are proposing a triangle (2D) based two-server PAKE protocol by incorporating trigonometry properties for signifying passwords. We are modifying the Yang et al. [23] model by fusing triangle properties with Diffie-Hellman key exchange to overcome the pitfalls of Yang et al. approach. Our protocol makes use of two servers, namely, Service Server(SS) and Control Server(CS); SS is solely for direct interaction with clients and CS stays at the back end. Client forwards the password as $g_2^P$ to SS. Server SS has stored value of $\omega$ (circumcenter of the triangle constructed from the password) whereas CS has password in the form of $\theta$ (angle between the medians of the triangle derived from the password). The two-servers never collude is the postulation made by existing PAKE protocols. The proposed protocol is unconditionally secure as credentials cannot be obtained even if the two-servers collude. A triangle is constructed from the user’s password to derive $\theta$ and $\omega$ and finally gets stored in SS and CS that can be used later for authentication. The protocol involves registration, authentication and key exchange phases. Initially the public parameters that are essential for registration and authentication are formed. The client needs to memorize solely the password for authentication. As a final point, during the key exchange phase the client and SS check out the equality of the secret key generated. During authentication and key exchange phase, communication is through a public channel that is more vulnerable to attacks.

In the initialization phase the public parameters essential for registration and authentication are distributed. Preceding to authentication client C picks out a password $P$ and creates authentication information. In the long run, the information to SS and CS are transmitted through secure channels. As a final point, the client creates a secret key analogous to server SS. During authentication and key exchange phase, the client and the two servers interconnect via a public channel.

The notation used in the 2D PAKE protocol are:

- $Zq^*$ - Group $\mathbb{G}$ of large prime order $q$
- $\mathbb{G}_1, \mathbb{G}_2, g, g_4$ - Generator of group $\mathbb{G}$
- $p$ - A large prime number
- $x_1, x_2$ Private keys $\in RZ_q^*$
- $y_1, y_2$ Public keys
- $b_2, b_3, a_2, a_4, r, r_1, r_2 \in RZ_q^*$
- $P$ - User’s password
- $b_4 = b_3 \oplus Hash(P)$
- $K$ - Secret key
- $\theta$ - Angle between the medians of the tetrahedron
- $c$ - Computed angle between the medians of the tetrahedron
- $s$ - Stored angle between the medians of the of the tetrahedron
- $\omega$ - Circumcenter of the tetrahedron
- $oc$ - Computed circumcenter of the tetrahedron
- $ocs$ - Stored circumcenter of the tetrahedron

**3.1 Registration and authentication and key exchange module**

Client C registers with the server SS with a password $P$. On receiving $g_2^P$ the server SS ascertains $g_4^\omega$ and constructs a triangle. SS holds the angle between the medians $\theta$ and $b_3$ in its database. The circumcenter $\omega$ and $b_4$ are stored in CS where

$$b_4 = b_3 \oplus Hash(P).$$

The user memorizes merely the password $P$ for authentication and key exchange. The user instigates the authentication by forwarding the username and $g_2^P$ to the server SS where $P$ is client password. Upon receiving the parameters, SS builds the triangle from $g_4^\omega$ and calculates the angle between the medians $\theta c$ and circumcenter $oc$. It verifies the $\theta c$ with the stored value $\theta s$. Further server SS forwards the request message to the server CS. Upon receiving the message the server CS verifies the received $oc$ with the stored $os$. If the verification holds well, CS forwards V2 (i.e.,) $oc$ value to SS to check out the trustworthiness of CS. On the other hand the Server SS generates a secret key with password credentials and conveys the parameter $H$ to the client. At the forefront the client validates the server, after receiving the key generation parameters. Finally the client generates a secret key as portrayed in the figure 1.
**Fig 1. Authentication and key exchange module of 2D PAKE**

### 4. SECURITY ANALYSIS

#### Proof of correctness

**Proposed protocol is correct if session key K generated in client side is equivalent to session key K′ generated in Server SS side.**

**Proof:**

In client side key K is computed from $B = g_1^{b1+b2}$ and

$S_u' = (B)^a = (g_1^{b1+b2})^a = g_1^{a(b_1+b_2)}$

Key $K = Hash(S_u', V_1)$ where $V_1 = g_2^{bc}$ and $b_1 = b_2 + Hash(P)$.

In server side SS computes key $K'$ from $A = g_1^a$, $V_1, b_4, S_2 = A^{b2} = g_1^{ab2}$ and from $S_u = A^{b1}S_2$

$A^{b1}S_2 = g_1^{ab1}g_1^{b_2} = g_1^{(a+b_2)}$

Key $K' = Hash(S_u, V_1)$

Therefore $K = K'$. Thus the correctness of 2D PAKE is proved.

**Theorem 1: 2D PAKE protocol is vigor against passive attack under the decisional Diffie-Hellman (DDH) assumption**

**Proof:**

Assume intuitively a passive adversary A has controlled the server CS and pretend to be CS to find out the secret session key and password by observing all interactions between SS and C. The adversary obtains (Username, $g_2^p$) from client. However, the adversary cannot gain knowledge about the password as it becomes discrete logarithm problem. In a similar sense if A obtains $g_2^0$, $g_2^ω(i.e.,)V_2/V_1$ from M3/M4 it is impracticable for the adversary to ascertain the password. Also, it is infeasible to obtain password P from the circumcenter(ω) and the angle (θ). Correspondingly attaining value from messages is impossible, as it happen to be discrete logarithm problem. In addition, key K is calculated with the aid of $V_1$ and $b_4$ that is stored in server SS and CS respectively. Additionally, the hash function ‘Hash’ is said to be one-way. Thus the protocol is vigor against passive attack.

**Theorem 2: 2D PAKE protocol is robust against active attacks**

(i) **Server SS/Server CS controlled by an active adversary and trying to imitate as client C**

**Proof:**

Consider that an active adversary A controlling the server SS/server CS to imitate as client C: 

(a) If an active adversary implicitly modifies $g_2^p$ transferred in M1 with his/her arbitrary number, then the authenticity of client fails in server SS as per equation (1). This is because the server uses the pre-stored values for verification.

Verify $calc g_2^{bc} = stored g_2^{bc}$ in SS

Verify $calc g_2^{bc} = stored g_2^{bc}$ in CS (1)

Note that $θ$ and $ω$ are derived from $g_2^ω$. Therefore $S_u$ verification in SS side is not a success as the key cannot be established per equation (2).

$S_u = Hash(g_2^{s_u})(2)$

Theorem 1: 2D PAKE protocol is vigor against passive attack under the decisional Diffie-Hellman (DDH) assumption
If there is change in $S$ value as $S'$, then hash of password verification is not valid because of the modified value $S'$ as per equation (3)
\[
\text{Hash}(S'_u, 0) \oplus H = \text{?Hash}(P)
\]
where $H = \text{Hash}(S_1, 0) \oplus b_2 \oplus b_4$ and
\[
b_a = b_a \oplus \text{Hash}(P)(3)
\]
Thus the protocol is robust against active attacks. As an outcome, the adversary is not efficient in active attack and thus holds the proof.

(ii) Server CS controlled by an active adversary tries to counterfeit as server SS

**Proof:**
Consider that an active adversary $A$ controlling the server CS counterfeit as server SS and modifies the messages exchanged between the servers and the client.

a) If an active adversary implicitly modifies $g_2^\theta$ transferred in $M_1$ with his/her arbitrary number, then the authenticity of client fails in server SS as per equation (1) stated in the Theorem 2 - proof (i).

In our 2D protocol attaining password is impossible since password related information is stored as $g_2^\theta$ and $g_2^{\omega}$. Obtaining $\theta$ and $\omega$ from $g_2^\theta$ and $g_2^{\omega}$ is considered as discrete logarithm problem.

b) Assume the adversary is modifying $M_4$: $V_1$, $b_5$: $A$, $S_u$ values. In such a case, verification of $S_u = \text{Hash}(S_2^u)$ fails. Since proposed 2D protocol is balanced PAKE, a password related information gets associated with session key. Subsequently key gets calculated in server SS using $K = \text{Hash}(S_1, V_1 b_3)$. Note that $V_1$ is a credential stored in SS and is not known to SS. In side client, key calculated using $K = \text{Hash}(S'_u, V_1 b_3)$ after the verification process of $(\text{Hash}(S'_u, 0) \oplus H = \text{?Hash}(P))$. Thus it overcomes the weakness of Yang et al. [23] protocol of cleaning the password and key.

**Remark 1:**
Therefore, identification of key generated between SS and C by an active adversary who pretends to be CS is unsuccessful. Thus the proposed protocol overcomes the drawback of Yang et al.[23] protocol.

(iii) Server SS controlled by an active adversary and tries to imitate as server CS

**Proof:**
Consider that an active adversary $A$ intentionally compromised the server SS to imitate server CS by modifying the messages exchanged between the servers and the client.

a) If an active adversary implicitly modifies $g_2^\theta$ transferred in $M_1$ with his/her arbitrary number, then the authenticity of client fails in server SS as per equation (1) stated in the Theorem 2 - proof (i).

b) Suppose the adversary modifies $g_2^{\omega}$ values, authentication fails in SS side as per equation (1), since $g_2^{\omega}$ is stored in SS and is not known to SS.

c) Similarly, if there is a change in $V_2$ value, authentication of client and CS fails in SS side as per equation (4).

calc $g_2^{\omega c} = \text{received} \ g_2^{\omega c}(4)$

d) If the adversary tries to modify the messages $M_3$, $M_4$, $M_5$, $M_6$ and $M_7$, verification of $S_u = \text{Hash}(S_2^u)$ fails internally. Also modification of values $m$ and $M_8$ results in authentication failure in client side as given in equation (3). In SS side, key computation involves $b_a$, a value stored in CS and SS has no knowledge about it. Therefore key identification by active adversary is unsuccessful.

**Remark 2:**
Thus, it overcomes the weakness of Yang et al.[23] protocol of determining the password and key by SS. Therefore key verification between SS and C by an active adversary who pretends to be CS is not successful. Thus the proposed protocol overcomes the drawback of Yang et al. protocol. This proves that 2D protocol is highly secure comparing to existing two-server PAKE protocols when both the servers’ database is controlled by adversaries.

**Claim 1:** Outside adversary counterfeiting a legitimate server is unsuccessful.

Intentionally, consider an adversary $A$ tries to imitate one/both of the servers, to get hold of the secret key computed between C and SS. The server SS computes $g_2^\theta$ and $g_2^{\omega}$ and place it in server SS and CS. Therefore, key verification between SS and CS is unsuccessful. Validation is unsuccessful when $g_2^\theta$ and $g_2^{\omega}$ are some arbitrary numbers chosen by the adversary. This concludes that impersonation by an outside adversary is not successful.

**Claim 2:** Instigating an offline dictionary attack by attaining access to the server’s database is unsuccessful.

Security is ensured by storing the encrypted $\theta$ value as $g_2^\theta$ in server SS, and similarly $\omega$ value as $g_2^{\omega}$ in server CS, respectively. Also, determining $\theta$ and $\omega$ from $g_2^\theta$ and $g_2^{\omega}$ is impossible as it is a discrete logarithm problem and thus, no valuable information can be obtained. If the adversary get hold of $\theta$ and $\omega$, finding the vertices of the triangle from $\theta$ and $\omega$ values is not practical[26], thus assuring three levels of security reasonably.

**Claim 3:** 2D PAKE protocol guarantees perfect forward secrecy of the established session key

In 2D protocol, key generation makes use of arbitrary numbers $a$, $b_1$ and $b_2$ associated with the two-servers and the client as shown in equation (5). As a result, it is impossible for an adversary to obtain the secret session key $K$ without implicit knowledge of the server’s and the client’s arbitrary number. Hence, the protocol guarantees forward secrecy

\[
K = \text{Hash}(S'_u, V_1^{b_3})
\]

where $S'_u = (B)^a = g_1^{a(b_1 + b_2)}$ (5)

**Claim 4:** 2D PAKE protocol ensures robustness against man-in-the-middle attack

If an adversary is able to counterfeit both the servers and client, at that point of time man-in-the-middle attack is possible. Here client authentication is confirmed in the server side as shown in equation (1) and server authentication is confirmed in the client side as shown in the equation (3).

Since mutual authentication takes place between the client and server, the protocol is strenuous against man-in-the-middle attack.

**Claim 5:** Protocol confrontation against online dictionary attack

It is restricted by the number of incorrect logins for a specific user account. As far as our protocol is concerned, user account locked automatically after 3 attempts. Thus the protocol is resistant to online dictionary attack.
Claim 6: 2D PAKE protocol ensures known-key security
Attaining session key $K'$ from previous session key $K$ is not possible. Since key generation makes use of arbitrary numbers $a, b_1$ and $b_2$ generated by the two servers and the client as shown in equation (5), it is infeasible to differentiate $g_1^{a(b_1+b_2)}$ from $g_1^{a_1(b_1+b_2)}$, $g_2^{b_1}$ generated for every session. Hence, it is impracticable to derive the session key $K'$ from $K$. Thus the proposed protocol confirms known-key security property as shown in equation (6).

$$K = \text{Hash}(S_y \cdot V_y^{b_2}) = \text{Hash}\left(\left(A^{b_1} \cdot A^{b_2}\right) \cdot g_2^{b_1}\right) = \text{Hash}\left(g_2^{a_1(b_1+b_2)} \cdot g_1^{b_2 \cdot \text{Hash}(P)}\right)$$

Claim 7: 2D PAKE protocol assures key control property
Key control property is neither of the interacting parties should compel the other secret key generation process. As per equation (5), the secret key is generated using the arbitrary numbers $a, b_1, b_2$ chosen by the client, server SS and CS respectively. None of the parties are compelling other and thus the claim assures key control property.

5. PERFORMANCE ANALYSIS
5.1 Communication performance in terms of bits
The communication complexity of SS is given by 12L+21, CS is given by 7L+11 and client is given by 4L+11 where L - number of group elements used in communication and l - number of hash values used in communication.

5.2 Communication performance in terms of rounds
From figure 1, it is apparent that the client engages with SS in 4 communication rounds, server SS engages with clients and CS in 8 rounds and server CS engages with SS in 4 rounds.

5.3 Computation performance
As per figure 1, 2D PAKE protocol has a computational complexity of 10 in client side, 15 in server SS side and 4 in the server CS side. As an outcome, it is perceived that the overall computation of the protocol is 29 which is to be valued. Number of computations in the proposed protocol is found to be lesser than the existing protocols.

Table 1. Complexity analysis of two-server PAKE protocols

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Client: Communication (bits)</td>
<td>4L + 11</td>
<td>2L + 21</td>
<td>6L + 21</td>
<td>&gt;15L</td>
</tr>
<tr>
<td>Client: Communication (rounds)</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Client: Computation</td>
<td>10</td>
<td>7</td>
<td>12</td>
<td>&gt;20</td>
</tr>
<tr>
<td>Server SS: Communication (bits)</td>
<td>12L + 21</td>
<td>6L + 3l</td>
<td>11L + 3l</td>
<td>&gt;19L</td>
</tr>
<tr>
<td>Server SS: Communication (rounds)</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Server SS: Computation</td>
<td>15</td>
<td>15</td>
<td>19</td>
<td>&gt;26</td>
</tr>
<tr>
<td>Server CS: Communication (bits)</td>
<td>7L + 11</td>
<td>4L + 11</td>
<td>5L + 11</td>
<td>&gt;19L</td>
</tr>
<tr>
<td>Server CS: Communication (rounds)</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Server CS: Computation</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>&gt;26</td>
</tr>
<tr>
<td></td>
<td>Rounds: 8</td>
<td>Rounds: 8</td>
<td>Rounds: 6</td>
<td>Rounds: 7</td>
</tr>
<tr>
<td></td>
<td>Client – S1 – S2</td>
<td>Client – S1 – S2</td>
<td>Client – S1 – S2</td>
<td>Client – S1</td>
</tr>
</tbody>
</table>

Table 2 recapitulates the primary security functionalities to be satisfied by the 2D PAKE protocol.

Table 2. Functionality comparison of 2D PAKE with Yang et al. protocol

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Yang et al Protocol [23]</th>
<th>2D PAKE protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known key security</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Forward secrecy</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Impersonation attack by passive/active adversary</td>
<td>Possible</td>
<td>Not Possible</td>
</tr>
<tr>
<td>Impersonation attack by outside adversary</td>
<td>Not Possible</td>
<td>Not Possible</td>
</tr>
<tr>
<td>Offline dictionary attack to server database to disclose the key</td>
<td>Possible</td>
<td>Not Possible</td>
</tr>
<tr>
<td>Key control</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Key size as per NIST recommendation &amp; OWASP project</td>
<td>3072-bit (RSA based key)</td>
<td>3072-bit (RSA based key)</td>
</tr>
<tr>
<td>SQL injection attack</td>
<td>-</td>
<td>Restricted</td>
</tr>
<tr>
<td>Man in the middle attack</td>
<td>Possible</td>
<td>Not Possible</td>
</tr>
</tbody>
</table>

DISCUSSION
1. As remarked earlier to preclude single point of vulnerability two-server password system has been proposed. However, all existing solutions for two-server PAKE architecture models presume that the two servers never collude to
disclose the password. To prevail over the assumption made by all existing two-server PAKE protocols, we proposed and proved that 2D protocol is unconditionally secure enough to withstand against offline dictionary attacks. (i.e.,) even when both the servers’ database is controlled by adversaries, it is not possible to glean the password as rightly pointed and proved in Theorem 2 and in Claim 2.

2. The protocol has been tested with Sqlmap, a database information takeover tool to prove the vigor of the proposed protocol against offline dictionary attacks.

3. In a similar fashion, 2D PAKE has been tested with Wireshark, Havij, Vega, Websecurity, Webcruiser, SSLSmart, WSAttacker and WSDigger to affirm the audacity of the protocol.

6. CONCLUSION

A well-formed 2D PAKE protocol using triangle properties is presented in this paper with formal proof. To overcome the drawbacks of Yang et al. protocol, 2D PAKE is designed under the balanced PAKE model. Remarkable characteristic is, server spoofing and obtaining password is not possible. This guarantees the strength of the protocol as discussed in the security analysis section. 2D PAKE protocol functions better than the existing two-server PAKE protocols as shown in table 1 and 2.

REFERENCES